

TypeScript Meetup v0.2

10 min	Around the room introductions
10 min	Opening comments - Anders Hejlsberg
20 min	TypeScript in the trenches - Eric Gamma
20 min	TypeScript update - Luke Hoban
20 min	TypeScript Q&A session - Luke, Anders, Steve Lucco, and Team
20 min	Break: Eat, drink, and Socialize
10 min	Bing and TypeScript - Kiran Badam & Sarvesh Nagpal
10 min	Cloud Make: Using TypeScript for Builds and WorkFlows - Wolfram Schulte
10 min	TypeScript language experiments - Ron Buckton
10 min	Xbox Music and TypeScript - Dan Rodgers (not recorded/streamed)
10 min	Using TypeScript to simplify JavaScript hosting - Paul Vick (not recorded/streamed)

Contact: [stevenic](#)

MICROSOFT CONFIDENTIAL

Thank You!

TypeScript Momentum

Community

Over 3000 CodePlex posts, 100 forks, 1000 StackOverflow questions, and 300 feature requests

Ecosystem

Over 180 .d.ts library definitions covering more than 90% of popular JavaScript frameworks.

Tool Support

IDEs: WebStorm, WebMatrix, JSBin, Web Essentials, Sublime Text, vi, Emacs, FlashDevelop, Brackets

Build: Heroku, Ruby, grunt, ASP.NET, node, compile-in-client

Testing: Chutzpah, tsUnit

Preprocessing: Bakehouse

Typings management: tsd

TypeScript 0.9

Generics

Generic classes, interfaces, and methods: The top requested feature by users

Overloading on Constants

Examples include createElement, getElementsByTagName, addEventListener

Other Features

Enum types, 'export =', function/module and class/module merging, methods in object literals

Compiler

Incremental parser, pull-model type checker, scaling to 100K+ line projects

Application Scale JavaScript

TypeScript in the Trenches

Erich Gamma

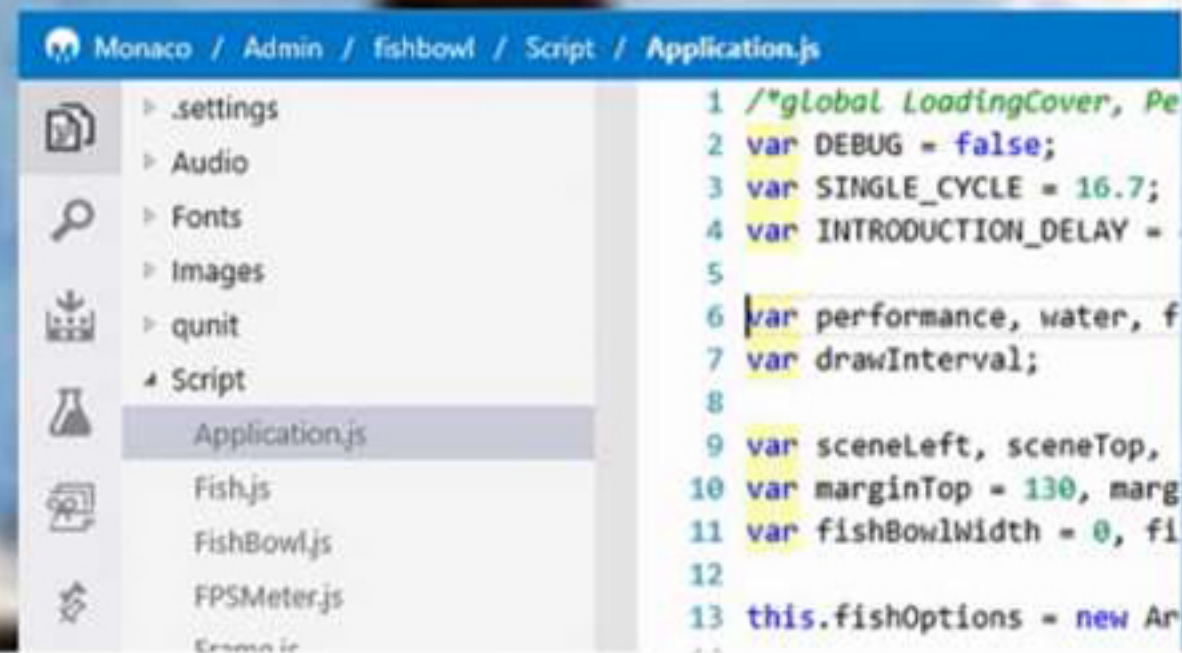
Microsoft

http://monacotools

[Try](#)[Download](#)[Learn](#)[Blog](#)[About Us](#)

Monaco Tools

Modern, web standards based developer tools,
platform, and cloud services



Easily get started

No long waits for installation and configuration.
Just log in and get coding!

Code in the cloud with confidence

Develop online with rich, contextual features like
IntelliSense, Source Assist, Quick Open and many
more!

Platform of reusable components

We've built Monaco as a set of simple, yet
extensible modules that you can adopt and
tweak yourself, based on your requirements and
scenarios

Team Foundation Server

Team Foundation Server 2012 / BFT_Projects

HOME WORK SOURCE BUILD TEST

explorer **changesets** shelvesets

Search work items

Change set number: Go Find

Recent changes

My recent changes

Changeset 25255

- BFT_Projects\Monaco\client\...
- controls
 - quickopen
 - commandHandler.js
 - commandHandler.str
 - gotoSymbolHandler.js
 - gotoSymbolHandler.str
 - openTypeHandler.js
 - openTypeHandler.str
 - search
 - openFileHandler.js
 - openFileHandler.str
 - ui\quickopen
 - quickOpen.js
 - quickOpen.str

gotoSymbolHandler.js - (Changeset 25255)

contents history **compare**

Previous File Previous Difference Next Difference Next File

gotoSymbolHandler.js; changeset 25233

gotoSymbolHandler.js; changeset 25255 (Latest)

```
1/3      return outinesupport && types.isFunction(outinesupport.getuut;
172      }
173      }
174      return false;
175      };
176      GotoSymbolHandler.prototype.autoSelectFirstEntry = function (searchValue) {
177      return searchValue.length > 0;
178      };
179      GotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchV
180      var results = [];
181      var searchValueRegex = new RegExp(strings.convertSimple2RegexPattern()
182      // Flatten
183      var flattened = [];
184      if(outline) {
185      this.flatten(outline, flatt
186      }
187      // Convert to Entries
188      for(var i = 0; i < flattened.le
189      var element = flattened[i];
190      var highlights = this.getH
191      if(highlights) {
192      // Decorate functions/w
193      var label = element.lab
194      if((element.type === 'm
195      label = strings.for
196      }
197      // Show parent scope as
198      var description = null;
```

stspioneer.8080/tfs/DevDiv_Projects/BFT_Projects/_versionControl/changeset?id=25255#path=

← → http://nickkirc-vm:8080/tfs/DefaultCollection/tfs/_versionControl/changeset/4#path=%2F Changeset 4 / Program.cs [...]

Changeset 4 / Program.cs [add]

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;

6
7 namespace HelloWorldConsole
8 {
9     class Program
10     {
11         static void Main(string[] args)
12
13         {
14             Console.WriteLine("Hello World");
15         }
16     }
17 }
```

Good usings
Martin Woodward - less than a minute ago - reply

Line level comment
Martin Woodward - less than a minute ago - saving...

dafny^{Microsoft} Research

Is this program correct? Ask dafny!

```
1 function Ackermann(m: int, n: int): int
2   // The following lexicographic pair allows Dafny to prove termination.
3   // Still, you may not want to sit around and wait for a call to Ackermann
4   // to terminate.
5   decreases m, n;
6 {
7   if m <= 0 then
8     n + 1
9   else if n <= 0 then
10    Ackermann(m - 1, 1)
11  else
12    Ackermann(m - 1, Ackermann(m, n - 1))
13 }
14
```

ask dafny

[home](#)

[tutorial](#)

[video](#)

[permalink](#)

Dafny program verifier finished with 1 verified, 0 errors

[samples](#)

[Hello](#)

[Fibonacci](#)

[CountToN](#)

[about Dafny](#) - A language and program verifier for functional correctness

Dafny is an imperative, object-oriented programming language with classes and inductive datatypes, and specification constructs for describing intended behavior. The Dafny verifier checks that programs live up to their specifications.

TypeScript Playground

TypeScript

learn

play

get it

run it

join in

TypeScript

JavaScript

```
1 class Greeter {
2
3   greeting: string;
4
5   constructor(greeting: string) {
6     this.greeting = greeting;
7   }
8
9   greet() {
10    return "<h1>" + this.greeting + "</h1>";
11  }
12 }
13
14 var greeter = new Greeter("Hello, world!");
15 var str = greeter.greet();
16 document.body.innerHTML = str;
17
```

greet () => string

() => string

greeting

```
1 var Greeter = (function () {
2   function Greeter(greeting) {
3     this.greeting = greeting;
4   }
5   Greeter.prototype.greet = function () {
6     return "<h1>" + this.greeting + "</h1>";
7   };
8   return Greeter;
9 })();
10 var greeter = new Greeter("Hello, world!");
11 var str = greeter.greet();
12 document.body.innerHTML = str;
13
```

Azure

Windows Azure

PREVIEW

egamma@microsoft.com

users

users

users preview

BROWSE SCRIPT COLUMNS PERMISSIONS

OPERATION

Insert

```
1 function insert(item, user, request) {  
2   if (item.userId !== user.userId) {  
3     request.respond(statusCodes.FORBIDDEN, 'You may only insert records with your userId.');4   } else {  
5     request.execute();  
6   }  
7 }
```

Office 365 - Napa

Office 365 • Napa • bubbles • App.js

Erich Gamma

- Content
- Images
- OfficeDocuments
- Pages
- Scripts
 - App.js
 - bubbles.txt
 - processing-1.0.0.js

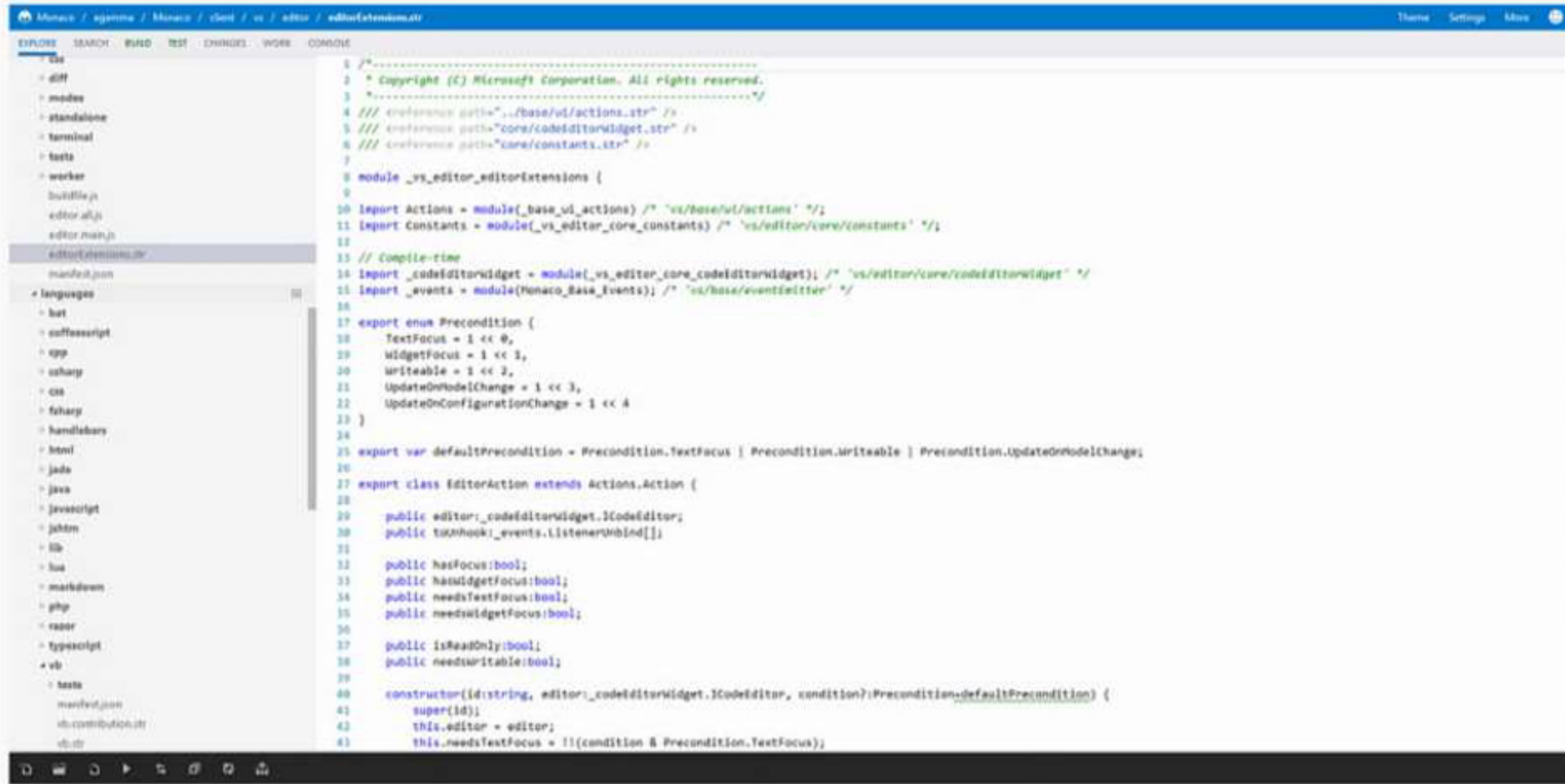
```
1 'use strict';
2 /*global Processing*/
3 var _values = [];
4 var cachedvalues = [['MSFT (cached)',27.34],['YHOO (cached)',19.85],['FB (cached)', 29.88]];
5
6 // The initialize function must be run each time a new page is loaded
7 Office.initialize = function(reason) {
8     $(document).ready(function() {
9         $('#read').click(getDataFromSelection);
10        $('#write').click(writeDataToSelection);
11    });
12 };
13
14 // Reads data from current document selection
15 function getDataFromSelection() {
16     Office.context.document.getSelectedDataAsync(Office.CoercionType.Matrix, function(result) {
17         Office.context.
18             if (result.stat contentLanguage string
19                 var yql = displayLanguage
20                 var queryUR document
21                 $.getJSON(q license
22                 var sto
23                 if (res
24                 var
25                 quo
26                 $.e
27
28                 });
29                 _va
30             } else {
31                 _values = cachedvalues;
32             }
33             //Display bubbles visualization
34             showCanvas();
35         });
36     });
37 }
```

result.value.join() + '\n';
'&format=json&env=http%3A%2F%2Fdatatables.org%2Fal

	A	B	C	D	E	F	G	H	I	J	K
1	MSFT										
2	GOOG										
3	FB										
4	YHOO										
5	APPL										
6											
7											
8											
9											
10											
11											
12											
13											
14											
15											
16											
17											
18											

A visualization of bubbles representing stock data. Two large green bubbles are visible: one labeled 'Google Inc.' and another labeled 'Apple'. The bubbles are positioned over a grid that corresponds to the spreadsheet data above.

Monaco @ Work - Dog Fooding



The screenshot displays the Monaco Editor interface, which is a code editor used in Visual Studio. The interface is divided into three main sections: a sidebar on the left, a top toolbar, and a main editor area.

Sidebar: The sidebar on the left shows a file explorer view. It contains a tree structure of files and folders. The 'languages' folder is expanded, showing various language-specific files like 'bat', 'coffeescript', 'cpp', 'csharp', 'css', 'fsharp', 'handlebars', 'html', 'jade', 'java', 'javascript', 'jshim', 'less', 'lua', 'markdown', 'php', 'razor', 'typescript', and 'vb'. The 'editorExtensions.ts' file is selected and highlighted.

Top Toolbar: The top toolbar contains several icons for navigation and editing, including 'EXPLORE', 'SEARCH', 'RUN', 'TEST', 'CHANGES', 'WORK', and 'CONSOLE'. On the right side of the toolbar, there are links for 'Theme', 'Settings', and 'More'.

Main Editor Area: The main editor area displays the content of the 'editorExtensions.ts' file. The code is written in TypeScript and defines the 'EditorAction' class, which extends the 'Actions.Action' class. The code includes comments, imports, and a constructor that initializes the 'EditorAction' class with a specific precondition.

```
1 /*-----  
2  * Copyright (C) Microsoft Corporation. All rights reserved.  
3  *-----*/  
4 /// reference path: ../base/ui/actions.ts  
5 /// reference path: core/codeeditorwidget.ts  
6 /// reference path: core/constants.ts  
7  
8 module _vs_editor_editorExtensions {  
9  
10 import Actions = module(base_ui_actions) /* 'vs/base/ui/actions' */;  
11 import Constants = module(vs_editor_core_constants) /* 'vs/editor/core/constants' */;  
12  
13 // Compile-time  
14 import codeeditorwidget = module(vs_editor_core_codeeditorwidget); /* 'vs/editor/core/codeeditorwidget' */  
15 import _events = module(Monaco_Base_Events); /* 'vs/base/eventEmitter' */  
16  
17 export enum Precondition {  
18     TextFocus = 1 << 0,  
19     WidgetFocus = 1 << 1,  
20     Writable = 1 << 2,  
21     UpdateOnModelChange = 1 << 3,  
22     UpdateOnConfigurationChange = 1 << 4  
23 }  
24  
25 export var defaultPrecondition = Precondition.TextFocus | Precondition.Writable | Precondition.UpdateOnModelChange;  
26  
27 export class EditorAction extends Actions.Action {  
28  
29     public editor: codeeditorwidget.ICodeEditor;  
30     public _hook: _events.ListenerUnbind[];  
31  
32     public hasFocus: bool;  
33     public hasWidgetFocus: bool;  
34     public needsTextFocus: bool;  
35     public needsWidgetFocus: bool;  
36  
37     public isReadOnly: bool;  
38     public needsWritable: bool;  
39  
40     constructor(id: string, editor: codeeditorwidget.ICodeEditor, condition: Precondition, defaultPrecondition) {  
41         super(id);  
42         this.editor = editor;  
43         this.needsTextFocus = !!((condition & Precondition.TextFocus));
```


Our Journey

	patterns		
	Small 50 kLOC	Medium 100 kLOC	Larger 200 kLOC
TypeScript	Modules Classes Interfaces Promises 10% Typescript	"AMD" Lazy Loaded Contributions 50% Typescript	Components Dependency Injection 100% Typescript
	2011	2012	2013

Inside

Classes

```
interface IRenderer {  
    getHeight(tree: ITree, element: any): number;  
    render(tree: ITree, element: any): void;  
}
```

Interfaces at Work

Options, JSON

```
interface IOptions {  
    hideButtons?:bool;  
    okOnFocusLost?:bool;  
}
```

Interfaces at Work

Functions, Callbacks

```
interface IMeasurementCallback {  
    (accessor: IMeasurementAccessor):void;  
}
```


Interfaces at Work

Indexed access

```
interface IEventMap {  
    [name: string]: EventListener[];  
}
```

Interfaces at Work

External Types

```
interface JQuery {  
  addClass(classNames: string): JQuery;  
  addClass(func:(index: any, currentClass: any)=>JQuery);  
  attr(attributeName: string): string;  
  attr(attributeName: string, value: any): JQuery;  
  ....  
}
```

Demo

Type definitions

TypeScript Type Definitions

<https://github.com/borisyankov/DefinitelyTyped>

amcharts	2 months ago	amCharts have been added [covobonomo]
angularjs	4 days ago	angular.d.ts ng.Module use of Object instead of {}, added tests [DerAlbertCom]
async	2 months ago	fix async-tests errors [Diullei]
backbone	a month ago	Optional parameters for Backbone. [Guuz]
bgiframe	19 days ago	Definition of bgiframe for IE6 - simple plugin implementation without... [sumegizoltan]
bootbox	3 months ago	Conformed tests to readme.md [vbortone]
bootstrap-notify	2 months ago	Added type definitions for bootstrap-notify [niemyjski]
bootstrap-datepicker	3 months ago	Update reference paths [borisyankov]
bootstrap	13 hours ago	Added overload for modal so can use backdrop: "static" [shearnie]
box2d	a month ago	Updated box2d, underscore, sugar [jbaldwin]
breeze	9 days ago	breeze updated - 1.2.7 [Diullei]
casperjs	2 months ago	casperjs header was normalized. [Diullei]
chai	2 days ago	Included Chai Expect. [kazimanzurrashid]
cheerio	2 months ago	Update readme and metadata for Cheerio [borisyankov]
chosen	3 months ago	Update reference paths [borisyankov]
chrome	3 months ago	Add type information for Chrome's Socket API [ahf]
codemirror	3 months ago	Update reference paths [borisyankov]
commander	a month ago	Add CommanderJs definitions [mdezem]
d3	11 days ago	Add more D3 Transition methods [rbirkby]
domo	2 months ago	Update domo definitions readme and metadata [borisyankov]

Our Journey

	patterns		
	Small 50 kLOC	Medium 100 kLOC	Larger 200 kLOC
TypeScript	Modules Classes Interfaces Promises	"AMD" Lazy Loading	Components Dep. Injection
	10% Typescript	50% Typescript	100% Typescript
	2011	2012	2013

Growing Pains

Managing scripts and their order

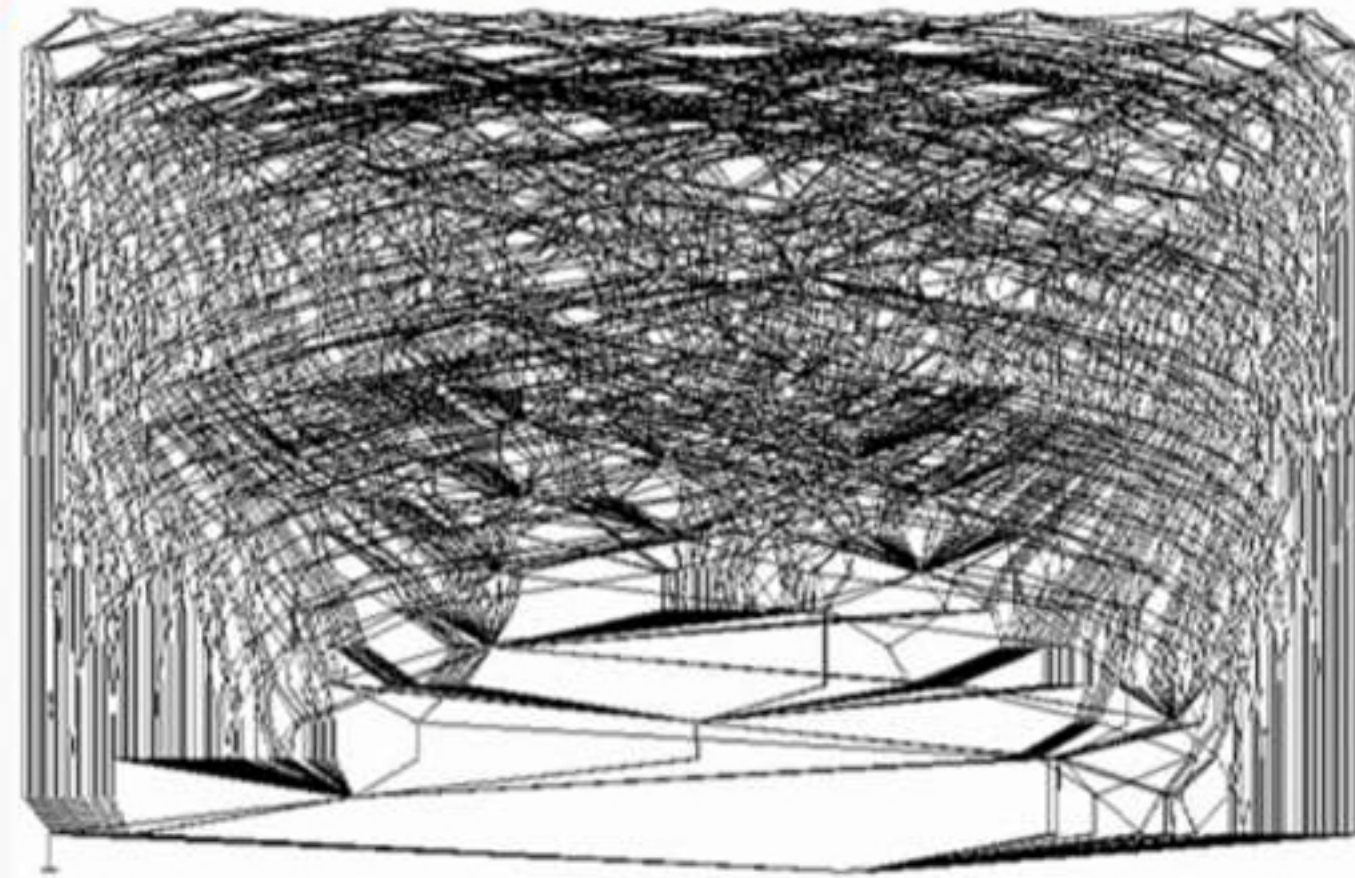
Manually edited lists of scripts

```
/ide/manifest.json
27
28   "src": {
29     "/monaco/ide/public/js/workbench.js": [
30       "public/js/constants.js",
31       "public/js/platform.js",
32       "public/js/events.js",
33       "public/js/core/context.js",
34       "public/js/core/storage.js",
35       "public/js/core/history.js",
36       "public/js/model/workspaceModel.js",
37       "public/js/ui/sitelet.js",
38       "public/js/ui/commands.js",
39       "public/js/ui/memento.js",
40       "public/js/ui/layout.js",
41       "public/js/ui/actions/action.js",
42       "public/js/ui/menu.js",
43       "public/js/ui/viewlets/viewlet.js",
44       "public/js/ui/parts/part.js",
45       "public/js/ui/parts/sidebarPart.js",
46       "public/js/ui/parts/editor/baseEditor.js",
47       "public/js/ui/parts/editor/editorModel.js",
48       "public/js/ui/parts/editor/editorInput.js",
49       "public/js/ui/parts/editor/editorOptions.js",
50       "public/js/ui/parts/editor/textEditor.js",
51       "public/js/ui/parts/editor/stringEditorModel.js",
52       "public/js/ui/parts/editor/stringEditorInput.js",
```


Growing Pains: Dependencies...

It is easy to import an internal module...

It is easy to contribute to a module...



"our dependency graph was such a mess that each area had a dependency on just about every other area." –Nick

AMD to the Rescue

define(id?, dependencies?, factory)

```
define([  
    'vs/base/lib/winjs', 'vs/editor/zoneWidget'],  
    function(WinJS, ZoneWidget) { ... }  
);
```


TypeScript: External Modules

TypeScript supports code generation for different module systems

```
import widget= import('vs/base/widget');
```

```
import http= import('http');
```

```
tsc --module amd app.ts    define([...], function(...) {...}
```

```
tsc --module commonjs HttpServer.ts    var http= require("http");
```

Before/After

AMD in JavaScript

```
define(['../winjs.base', '../zoneWidget'],  
    function(WinJS, ZoneWidget) { ... }  
);
```

TypeScript

```
import WinJS= import('vs/base/lib/winjs');  
import ZoneWidget = import('vs/editor/zoneWidget');
```

AMD Applied – Self Contained Modules

Support à la carte consumption

Expressing CSS dependencies

CSS AMD loader plugin

TypeScript pragma

```
///amd-dependency path="vs/css!./actionbar" />
```


Growing Pains – Self Contained Modules

Start-up time

Eager loading of scripts and CSS

Support à la carte consumption




















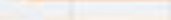








Slow startup: #requests, data transfer

Expressing CSS dependencies

CSS AMD loader plugin

TypeScript pragma

/// <amd-dependency>

Name Path	Method	Status Text	Type	Initiator	Size Content	Time Latency	Timeline	40ms	70ms
 version.js /public/	GET	200 OK	application/javascript	FPStylet.js:14 Parser	(from ca...)	35ms 20ms			
 base.js /public/webpack-core-editor/webpack-base.js	GET	200 OK	application/javascript	FPStylet.js:14 Parser	(from ca...)	39ms 31ms			
 strada.js /public/webpack-core-editor/webpack-strada.js	GET	200 OK	application/javascript	FPStylet.js:14 Parser	(from ca...)	37ms 31ms			
 javascript.js /public/webpack-core-editor/webpack-javascript.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	319B 22.25KB	355ms 40ms			
 vxamd.js /public/webpack-core-editor/webpack-vxamd.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	319B 8.15KB	311ms 40ms			
 editor.js /public/webpack-core-editor/webpack-editor.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	320B 401.35KB	188ms 40ms			
 jsbeautify.js /public/webpack-core-editor/webpack-jsbeautify.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	319B 44.15KB	360ms 40ms			
 javascript-vs.js /public/webpack-core-editor/webpack-javascript-vs.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	319B 3.25KB	362ms 51ms			
 css.js /public/webpack-core-editor/webpack-css.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	319B 14.44KB	622ms 60ms			
 strada.js /public/webpack-core-editor/webpack-strada.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	319B 9.72KB	619ms 60ms			
 php.js /public/webpack-core-editor/webpack-php.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	319B 22.55KB	621ms 60ms			
 html.js /public/webpack-core-editor/webpack-html.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	319B 22.14KB	625ms 60ms			
 markdown.js /public/webpack-core-editor/webpack-markdown.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	319B 6.70KB	635ms 67ms			
 csharp.js /public/webpack-core-editor/webpack-csharp.js	GET	304 Not Modified	application/javascript	FPStylet.js:14 Parser	319B 31.37KB	637ms 68ms			

Lazy Loading Contributions

[csharp.contribution.ts](#)

```
modeRegistry.registerMode(  
    ['text/x-csharp'],  
    new Platform.Descriptor(  
        'vs/languages/csharp/csharp',  
        'CSMode')  
    );
```

[csharp.ts](#)

```
export class CSMode extends  
modesExtensions.AbstractMode {  
    constructor() {  
        super('vs.languages.csharp');  
    }  
    // lots of code ....  
}
```



After the AMD Migration Impressions

"It feels like **fresh** showered. Self contained modules, no more cycles, no more globals, clean file system structure" -Alex

Our Journey

	patterns		
	Small 50 kLOC	Medium 100 kLOC	Larger 200 kLOC
TypeScript	Modules Classes Interfaces Promises	"AMD" Lazy Loading	Components Dep. Injection
	10% Typescript	50% Typescript	100% Typescript
	2011	2012	2013

100% TypeScript

"Writing JavaScript code in a large project is like **carving code in stone**" –Alex

Migration is code clean-up and real work

Velocity around 300 LOCs per hour

"As I did conversions, I began **typing various object literals** I was passing around as interfaces. Soon enough, **I realized how inconsistent I was, the same data was flowing around in at least 3 different formats**. This is because of the easiness through which you can create literals in JavaScript Need some placeholder for data?... Just create a new literal object." --Alex

Even More Growing Pains

“Writing JavaScript code in a large project is like **carving code in stone**” –Alex

Tooling slow down, translation time

Tools need to digest lots of source to infer types

Hard coded dependencies on context

Decreased testability

Difficulty to reuse components in different contexts

Componentization

Tooling slow down, translation time

Tools need to digest lots of source to infer types

Reuse TypeScript code as 'binary' JS components with a declarations file

Example using TypeScript language services as a component

Compiler and language services > 30kLOC of TypeScript

```
tsc --declarations --out typescriptservices.js typescript.ts
```

Dependency Injection

Dependencies on *services* provided by container

Decreased testability

Need for configuration flexibility

Services (20)

Selection, Progress, Logging, History, Events, Configuration, Telemetry, Requests, ...

Dependency Injection

```
export interface IProgressService {  
    show(total?:number, delay?:number):IProgressRunner;  
}
```

```
export interface IProgressServiceConsumer {  
    injectProgressService(service:IProgressService):void;  
}
```

```
export class Explorer implements Services.IProgressServiceConsumer {  
    private progressService:Services.IProgressService;  
    public injectProgressService(service:Services.IProgressService) {  
        this.progressService = service;  
    }  
    //....  
}
```


TS Retrospective

We were on the **bleeding edge**...

...but we expected it and had plenty of band aid

We would **do it again**, the benefits outweigh the pains

Code readability, refactoring agility, tooling, fun

We would start with TypeScript (and AMD) from the beginning

<http://www.typescriptlang.org/>